# A Combined Global-Analytical Quality Framework for Data Models

Lemaitre, Jonathan; Hainaut, Jean-Luc

*Published in:*
Proceedings of the 3rd International Workshop on Quality in Modeling (QiM'08 @ MODELS'08)

*Publication date:*
2008

*Document Version*
Early version, also known as pre-print

# A Combined Global-Analytical Quality Framework for Data Models

Jonathan Lemaitre and Jean-Luc Hainaut

Computer Science Faculty, FUNDP, Namur, Belgium
{jle,jlh}@info.fundp.ac.be

**Abstract.** In the domain of data model quality two independent approaches can be identified. The first one proposes a global view mainly based on quality models and frameworks, focusing on high level quality characteristics such as minimality, maintainability and evolvability and on metrics for measuring them. A second research track has concentrated for decades on the analysis of specific problems, ranging from unnormalized structures to unsatisfiability. The latter proposes means for formalizing, detecting and correcting particular defect patterns. Both of these approaches address data model quality issues, but in independent ways. In this paper, we present an attempt to address database schema quality through both approaches in a common framework. We summarize the main concepts and reasoning basis of a project devoted to database schema quality. We propose an operational framework that combines the contribution of both global and analytical views of quality. Our global view focuses on defects categories to evaluate schema quality and error side effect. Our analytical view translates into detection and correction methods of these defects. The final purpose of this work is to propose a precise, intuitive and easy to use quality management methodology for database schema.

## 1 Introduction

Quality has become one of the major topics in software engineering. Research and industrial communities acknowledge that such concepts as maintainability, portability or evolvability translate in technical terms users satisfaction and economical stakes. The question has been at the core of software engineering for more than three decades. In the nineties, authors have already assessed the impact of poor quality and errors made during the modeling phase [1]. During the last few years quality of models became more and more important owing to the increasingly popular MDE approaches mainly relying on modeling and models transformations.

Looking at data model quality, one can make the distinction between two research approaches. The first one comprises proposals allowing a global assertion of the schema quality through such key concepts as quality models, frameworks and metrics. Quality models are mainly composed of definitions of global quality characteristics [2, 3] while frameworks define particular views and/or contexts of use for the characteristics [4–6]. These authors define metrics that provide a numerical evaluation of the characteristics of a particular schema. On the other hand some authors study very specific problems as, for example, normalization [7–9], visualization [10, 11], satisfiability [12] and more generally the intra-model consistency [13, 14], etc. They propose a limited but formally defined view of model quality that includes precise problem identification techniques and problem solving.

As far as model quality is concerned, both research approaches have their advantages and limitations. The first one provides an abstract, fast but imprecise global evaluation of a model that can translate into, e.g., development and maintenance time and cost. The second one leads to a precise identification of intuitive classes of structural problems and to their solution, but is of no help when a global evaluation is required. Though they both address the problem of data quality problem, they have been so far developed independently of each other.

The goal of our research is to build a framework that relies on both global and analytical approaches. The result that we would like to obtain is a framework easily applicable and intuitive. This framework will propose methods and tools for addressing specific structural problems and assessing schema quality. Quality assessment will also be associated with particular correction methods. This framework will be valid for data models of future software systems as well as for models of existing, and even legacy, software systems.

Our research focuses on persistent data structures, that is, on database schemas. Though this proposal is independent of the schema abstraction level (i.e., it covers both PIM and PSM levels), we will base the discussion on conceptual examples expressed in a variant of the ER formalism [15].

This paper is structured as follows. First, we recall some aspects of global (section 2) and analytical (section 3) approaches. Section 4 presents our proposal for unifying the these approaches. In section 5 we illustrate our approach with an elementary study of a specific quality characteristic, namely understandability. The last section (6) includes first conclusions and future work.

## 2 Global approaches of schema quality

In this section, we address informally the main characteristics of the global approaches to schema quality. Quality frameworks and models can be classified into identifiable categories:

- Quality models: they may also be viewed as hierarchical frameworks since they propose a tree-structured view of the quality. The root concept is the global quality which is divide into different global aspects of the quality. Each of these global aspects can also be refined into more specific aspects. The leafs are generally associated with metrics. One of the most famous example is the ISO9126 standard [2]. In this standard, the first level is the quality. The second level copes with such characteristics as maintainability or efficiency. The third level details sub-characteristics depending on one

characteristics, e.g., stability and analyzability contributing to maintainability. In the domain of the database schema, similar hierarchical models have been proposed, e.g., by Hoxmeier [16]. Quality models may also be associated with other kinds of frameworks. For example, some quality models express the links between the characteristics [17, 4] or the relationships between the characteristics and the actors of the modeling process [4]. Quality models have the advantage to propose a structured view, but as quality terms are often at a high abstraction level, they may prove difficult to apply. Furthermore definitions often are ambiguous due to the lack of agreement on the meaning of essential terms.
- Formalization framework: frameworks of this category are quite uncommon. They propose a methodology for building quality metrics using mathematical properties the metrics have to satisfy [18, 19]. Thus these frameworks may not provide quality definition, but a way to enhance the validity of new metrics.
- Causality framework: rather than proposing a hierarchical view of quality, some authors highlight the influence of some quality characteristics on others [17, 5, 20]. This kind of frameworks seems to have more practical use than the others but often require costly empirical validation that provides precise numerical coefficients that measure the characteristics correlation. Moreover, these models address a limited number of quality characteristics. In this category, we can mention the work of Kesh [17] and Maes and Poels [5].
- Semiotic framework: these proposals started with the work of Lindland and al. [21]. They are designed for conceptual model in general and they are rooted in the study of sign processes and detail quality into syntactic, semantic and pragmatic aspects. Later, Krogstie made a new proposal [6] integrating additional aspects such as physical quality and social quality. These frameworks give a *constructivist world-view* [6], i.e., they represent the situation of quality aspects with the related actors of the modeling process (e.g., model, language, user). Other proposals were made occasionally but are very close to the framework of Krogstie [4, 22]. These frameworks have the advantages to represent the modeling context and to link the quality with it. Unfortunately they also stay at a very theoretical level without proposing easily usable means for assessing the quality.

As main advantages of quality frameworks, we can underline the global assessment of the quality and the structuring of the reasoning induced by the frameworks. Nevertheless quality characteristics have different meaning across different frameworks. The frameworks also stay at a theoretical level that impairs their understandability and/or usability.

Considering the data model domain, several metrics are available. Among the different proposals the complexity of the metrics expression may vary from simplistic to overly complex. Metrics are based on counting particular schema objects such as entity types, attributes, relationship types, is-a hierarchies, etc. Using empirical studies, the authors assess the value of some quality characteristics. These values are associated with the result of the simple object counts, which, for example, may be used to define linear or quadratic polynomial functions using objects count results as parameters. As example we may underline the work of Piattini et al. [23–25] concerning UML class diagrams metrics based on structural properties. Metrics may also be included in a more global view of quality represented by a quality model [26]. The simplest metrics functions are shaped as $\Sigma_{i=1}^{n} a_i x_i$, where $a_i$ is a coefficient and $x_i$ a simple objects count. The expression below presents an example of a more complex metrics, where $ASvsC$ is the Associations versus Classes metrics, $N^{AS}$ is the number of associations in an UML class diagram and $N^{C}$ is the number of classes [27]. Typically, the $ASvsC$ score has to lie between, e.g., 0.3 and 0.6. A result below 0.3 indicates a lack of relationships between classes while a result above 0.6 probably testifies to a lack of modularity (spaghetti-like schema).

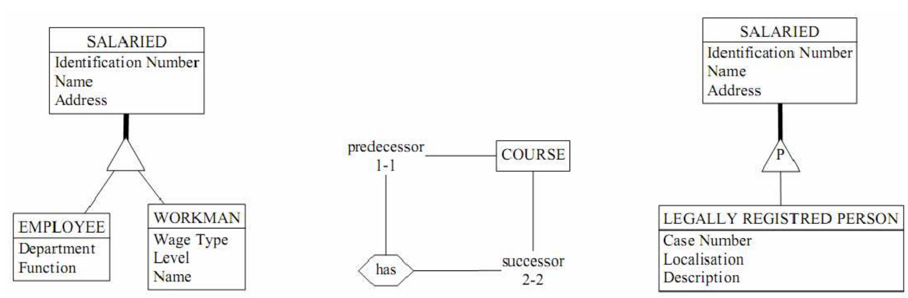
$$\boxed{ASvsC = \left(\frac{N^{AS}}{N^{AS} + N^{C}}\right)^2}$$

A metric may evaluate quality characteristics like the clarity or expressiveness [28] or low level properties like structural properties [27, 29]. A global overview of metrics is, as for the frameworks types, out of the scope of such a paper, but the following advantages and limitations may be expressed. Metrics are directly applicable and easily usable. However they often are unintuitive and are very costly to validate. This induces difficulties for interpreting results. Furthermore, metrics quality evaluations are based on occurrence frequencies of specific objects in a schema and the comparison with the authorized value intervals of these frequencies. Thus metrics give only a global quality result that hardly allows users to locate some precise defects they can correct in a schema.

## 3 Analytic approaches to schema quality

An analytic approach concentrates on specific types of defects. Such defects can be formally detected. Their harmfulness has been studied and correction techniques have been proposed. Many proposals have been made for the detection and the correction of specific defects located in models. Some of them may be general enough to concern most of the software product types, such as syntactic errors. Others concern only specific models types as for example the normalization which address data schema and was originally designed for the database logical schemata [7–9]. Defects can be classified according different aspects, e.g., syntax, semantics, readability or maintainability. In our work, we focus on the structural defects, that can be formally identified by schema analysis. The problems related to the application domain semantics have not been considered. We have also discarded visualization aspects [10, 11] (distance between objects, shape, color, etc.) and concrete syntax of the models (e.g., complying with the graphical convention of a specific editor) which can be dealt with independently. However, as data models imply a graphical representation, the visual aspect cannot be ignored. Its influence and the way it is taken into account will be mentioned in Sec. 5.

Two types of defects can be highlighted. The first includes the *normalization defects* which are *structures in the schema that suggests, and sometimes even scream opportunities for transformations, considering specific requirements like the readability, the evolvability, the performance, etc.*. This definition is derived from the bad-smells definition given by Mens and Demeyer [30]. The second type of defects are the *correctness defects* which comprises *the errors that prevent the schema to be translated into a physical schema or to meet users expectations.* Uninstantiable structures form an important class of errors: they are schema constructs for which there is no valid instances [13, 12]. As we focus on defects patterns, we may underline the recent work of Wahler [31]. He proposes a pattern approach for defining and detecting UML-OCL constraints inconsistencies. Normalization defects (ND) are awkward or inappropriate structures that don't make the schema incorrect or not instantiable. A change in the schema is not mandatory in opposition to the correctness defects (CD) that have to be corrected. Examples in figure 1 give an example of two CD and one ND. Schema (a) violates a syntactical rules stating that the super-type and one of its sub-type may not have attributes with the same name. Schema (b) contains a semantic error due to unsatisfiable cardinality constraints (the only finite population satisfying `COURSE` is empty). Schema (c) is correct but includes an is-a relation with only one sub-type but a partition constraint (symbol `P`). Normally, the super-type and sub-type should be merged since they have the same population.




(a) (b) (c)

**Fig. 1.** (a) ER schema containing syntax error. (b) ER schema containing a semantic error. (c) ER schema contains an abnormal construct.

Let us finally mention the class of *unexpressive* or *insufficiently expressive* structures. They are correct structures whose semantics could be better expressed through more appropriate structures. Two examples are very common, namely *attribute entity types* (simple properties expressed as entity types instead of attributes) and *relationship entity types* (associations expressed through entity types instead of relationship types). Unexpressive structures may be considered as ND that may reduce the conciseness of the schema, hence its readability, and finally its maintainability. Eick [32] propose a first approach on data schema understandability and its enhancement using transformations. Rauh and Stickel [33] also proposed a transformation based solution for normalizing ER schemas. Finally, Assenova and Johannesson also studied the schema readability and propose a solution for restructuring a schema based on transformations [34].

As compared with global approaches, which produces metrics based on frequencies and ratios of simple objects (entity types, attributes, relationship types, is-a links, ans so on), analytical approaches study object patterns that generally are complex and semantically rich, but without attempting to count them. In addition, these patterns are most often associated with correction transformations that can be used to improve the quality of the schema.

## 4 Proposal of a combined approach

The goal we have chosen to reach in this research is to design a quality evaluation and improvement framework for database schemas. In particular, by integrating the contributions of the global and analytical approaches, we expect (1) to augment global approaches with metrics based on semantically rich structural patterns considered as defects, (2) to associate with each structural pattern correction transformations that can be either suggested or automatically applied. By targeting precise structural defects, we expect more informative metrics which better describe the quality of the schema.

Practically, we have structured our work in three steps. The first step is the identification of defect families, formalized by generic patterns resulting from various domains of database theory such as relational normalization, conceptual normalization, satisfiability, redundancy techniques, empirical (good) practices,etc. The second step consists in integrating these patterns into global quality frameworks, hence improving existing metrics systems. The third step addresses quality improvement. This process mainly relies on transformational techniques [35].

The framework we are building is made up of four components.

– A defect taxonomy. Each of the families mentioned in Section 3 are decomposed into more specific categories. The *correctness defects* family comprises two categories, namely syntactical errors and inconsistent constructs. The *normalization defects* family includes seven categories: non minimal constructs, unexpressive constructs, abnormal constructs, irregular constructs, redundant constraints, redundant structures, internal redundancies, presentation defects and standard violation. Each defect of each category is precisely defined by a structural predicate with which the schema can be parsed to identify defect instances. Such predicates can be expressed in some sort of constraint language such as UML OCL [36] or through logic-based languages as described in [35]. In addition, each defect receives a practical documentation comprising an informal description, the conditions, the paradigm and the abstraction level of the schema where it generally appears as well as some representative examples.

- A limited set of quality characteristics. These characteristics are drawn from standard global approaches proposed in the literature. They are linked to the defects families and categories. The links indicates the influence without using precise ratio factors in order to keep the framework as simple as possible. The rationale of this influence is also explained.
- Assessment methods for the quality characteristics. For assessing the global value of a characteristic we propose a simple counting method based on weights. The weights are declared in the properties of the specific defects description. Assessment also includes a relative evaluation for equivalent structures. Considering a quality characteristic and two different but semantically equivalent structures for expressing the same concepts, the structure with the higher weight for the characteristic would be a better choice, all other weights being equal. A validation procedure is being experimented with the help of a limited team of database design experts. The experts are asked to sort semantically equivalent structures according to their preferences. This procedure seems to bring important advantages compared with usual global approach validation processes: the expert have to compare and to evaluate small structural patterns and not complete schemas, their evaluation is reusable since they are domain independent and finally, the requested effort is quite small (typically half a day).
- Correction methods for the defects detected. When it is possible, corrections methods and changes advises are proposed for improving schema quality. If the correction is obvious and there is only one possible choice, the change can be applied automatically. In the other cases, a list of solutions is propose to the users of the framework. Defects correction will rely on transformational techniques. Model transformations is one of the main baseline of the MDE approach and is known since years in the database domain [35]. This approach follows that of Assenova and Johannesson [34] though we also use non semantics-preserving operators.

## 5 First application on schema "understandability"

As a first illustration, we will discuss the concept of understandability. There is so far no agreement on a common definition even though the main idea is accepted by most authors. Table 1 collects some of the most popular definitions.

Those definitions are very abstract, so that the authors cannot provide detail about how to evaluate the understandability of a software artifact in general. Except for the last definition that refers to design and structure, the understandability may be considered under various contexts: adequate choice of name, appropriate visual organization, design complexity, etc. Understandability metrics may compute the total time required by the reader for understanding the schema. Unfortunately, this is an "a posteriori" metric. An "a priori" global evaluation is a lot more difficult to develop since human ability and experience differ among users.

**Table 1.** Understandability definitions

| Definition | Author(s) |
| --- | --- |
| A software requirement specification (SRS) is understandable if all classes of SRS readers can easily comprehend the meaning of all requirements with a minimum of explanation. Readers include customers, users, project managers, software developers, and testers. | Davis et al. [3] |
| The understandability is the capacity of the software product to be understood, learned, used and attractive to the user, when used under specific conditions. | ISO/IEC 9126 [2] |
| An SRS is understandable if all classes of SRS readers can easily comprehend the meaning of all requirements with a minimum of explanation. | Krogstie [6] inspired from Davis et al. [3] |
| Understandability is defined as the ease with which the concepts and structures in the data model can be understood. | Moody [4] |
| The properties of the design that enable it to be easily learned and comprehended. This directly relates to the complexity of the design structure. | Bansiya and Davis [26] |

In the framework we are developing, we consider the understandability of a schema, disregarding visual aspect, as follows: **We use the global definition of understandability given in the ISO/IEC 9126 norm [2]. We consider that the schema has to be understood by persons who are familiar with the modeling language and its best practices. Hence we consider as a factor of understandability, the adequacy of the schema constructs used with respect to such good practices.** In other words, when the language offers different constructs for expressing a definite concept or fact type, we suggest to use the construct with the highest weight of adequacy according to the reference expert team. Considering the analytical approach, we identified different categories of structures transmitting the same information but using different structural means for that. For each category, experts may associate an understandability score to the patterns. We illustrate this process using the Is-A Partition category.

Figure 2 gives four semantically equivalent but structurally different schemas. Those schemas express the following facts:

1. `A` has a `A1` and a `A2`;
2. `B` owns a `B1` and a `B2`;
3. `C` has a `C1` and a `C2`;
4. `A` is either a `B` or a `C`.

In schema (a), the fact 4 is expressed by an is-a hierarchy in which super-type `A` has two subtypes `B` and `C`. The is-a relationship is defined as a partition, represented with symbol `P`. In schema (b), the fact 4 is represented using relationship

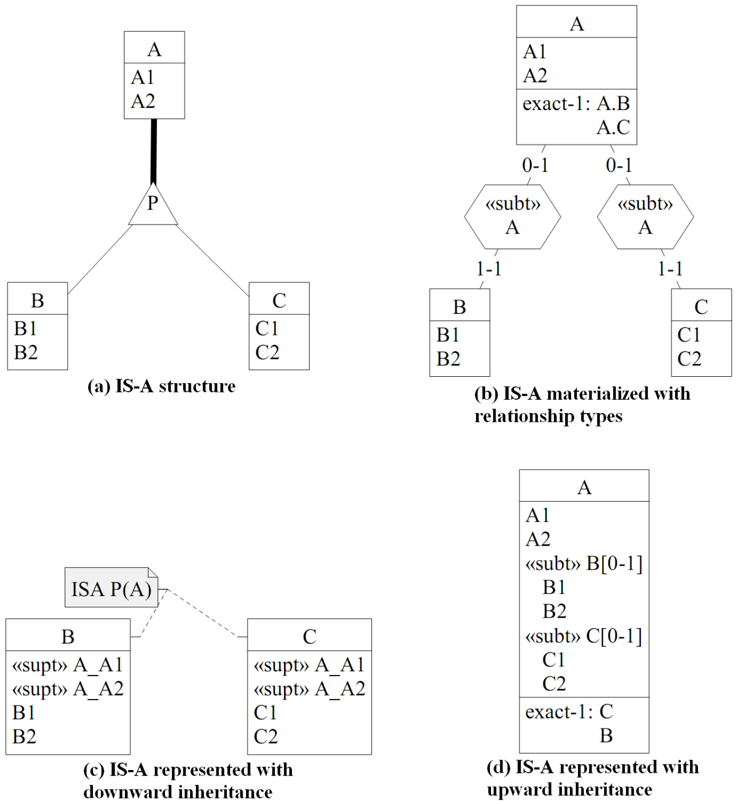



**Fig. 2.** 4 semantically equivalent structures of the is-a category.

types with the stereotype [1] *subt*, representing the is-a, and an "exactly-1" constraint standing for the partition. Schema (c) represents the information with the downward inheritance, meaning that `A` is materialized in `B` and `C`. The stereotype *supt* highlights the attributes of the supertype. The textual note indicates the type of the is-a. Finally, the schema (d) stands for the upward inheritance which materialized the subtypes into the supertype. The subtypes elements are marked with the stereotype *subt*. As in (b), the "exactly-1" constraint represents the partition.

Obviously, schema (a) complies with the best practices in conceptual modeling, while schema (b), (c) and (d) come closer to lower abstraction level models, e.g., the relational model. Those representations of the concept of category/subcategory have been evaluated respectively *very good*, *average*, *bad* and *average*

[1] Stereotypes are surrounded in the schemas by "<<" and ">>" signs

for (a), (b), (c) and (d) by our research team [2]. The used scale is composed of 5 values : *very bad*, *bad*, *average*, *good* and *very good*. This scale is also considered as an ordinal scale. Indeed, the difference of quality between two values is hardly quantifiable. Plus, the values are discreet and comparisons between values are authorized(*very bad* < *bad*, etc.).

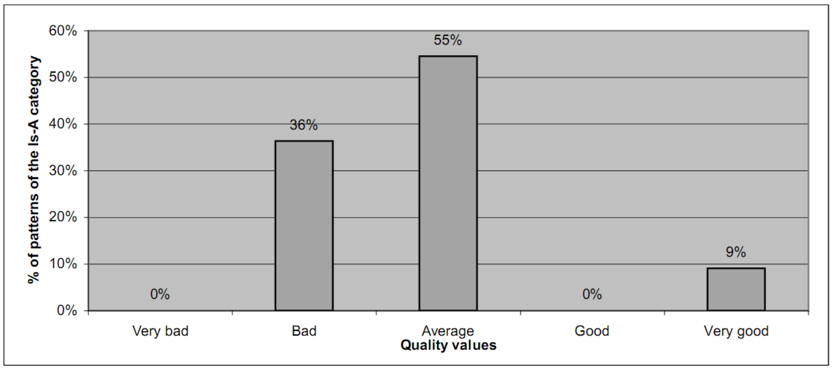



**Fig. 3.** Graphical representation of the results of the 1st metric applied on a fictive example.

Scores are used to compute the global understandability of the schema for the structures categories. The first metric will give the proportion of *very bad*, *bad*,...in the schema for a category. This metric respect the properties of the ordinal scale used for scoring. The metric highlights understandability problems, but with a composed result. Fig. 3 propose a graphical representation of possible results.

A second metric may be proposed by attributing numerical values to the scores, e.g. *very bad* = 0, *bad* = 1, etc. The weights are transformed to 4, 2, 0 and 2, respectively for (a), (b), (c) and (d) in Fig. 2. This allows us to compute a global understandability measure of any schema for a structures category: for each construct of the schema of the same category, we note the actual weight and the maximum weight. By summing the former on the one hand and the latter on the other hand, then by dividing the first sum by the second one, we get a global understandability measure in the range [0-1]. If the value of this average is close to 0, it indicates a poor quality, while a result close to 1 indicates a good quality. Obtained results are aggregated and easy to read. However, they are violating the ordinal scale properties.

Interestingly, there exist semantics-preserving transformations that produce schema of the Fig. 2 from each other. These transformations are triggered by the

[2] Our research team is composed of 4 people. Without taking into account the years of study, experience in the database field of the team is: one has 30 years of experience, another has 10 years and the 2 last have 5 years.

detection of an instance of a source pattern. By selecting the equivalent pattern with the highest weight, we can automatically fix bad smell defects.

## 6 Conclusion and future work

This paper introduces a quality framework based on specific data models defects. It derives from the merging of two independent approaches, namely global approaches based on metrics, and analytical approaches that study defect categories and their corrections.

By this framework, we improve the precision and the acceptability of global metrics. In addition, we make it possible, not only to evaluate quality characteristics of a schema, but also to improve them.

As shown in [35], transformations are completely specified by pre- and post conditions, so that they can be implemented in CASE tools. We have developed an extension of the DB-MAIN CASE tool which identifies defect patterns in a schema, and which suggests possible improvement.

This work started in early 2007, so that several problems and questions still have to be studied. We mention three of them, on which we are currently working:

- **What are the interactions between the different quality criteria?** As the causality frameworks express it, the quality criteria influence each other. This influence has to be made explicit. For avoiding this problem, we will try to obtain a limited set of criteria with disjoint view of quality.
- **May the improvement of a structure change the quality of adjacent structures?** Transformations of a structure may influence adjacent structures. It means that improving a part of the schema may decrease the quality of another part. This has to be made clear and detailed in the transformation properties.
- **Is the automated process a better choice?** One of the main goal of the quality discipline is to obtain automatable processes. However this is not realistic when design expertise is required. As we focus on formalizable problems, the patterns and transformations may be implemented into an modeling tool but some transformations choices will have to be selected manually. As an example, the correction of errors or the choice between transformation having equivalent quality results need human intervention.

## References


1. Standish Group: Chaos: A recipe for success. Technical report, Standish Group International (1999)
2. ISO/IEC: ISO 9126-1:2001, Software engineering - Product quality, Part 1: Quality model. ISO/IEC (2001)
3. Davis, A., Overmyer, S., Jordan, K., Caruso, J., Dandashi, F., Dinh, A., Kincaid, G., Ledeboer, G., Reynolds, P., Sitaram, P., Ta, A., Theofanos, M.: Identifying and measuring quality in a software requirements specification. In: Proceedings of the First International Software Metrics Symposium. (1993) 141–152

4. Moody, D.L., Shanks, G.G.: Improving the quality of data model: Empirical validation of a quality management framework. International Journal of Information Systems **28** (2003)
5. Maes, A., Poels, G.: Evaluating quality of conceptual models based on user perceptions. In: International Conference on Conceptual Modeling - ER 2006, Tucson, Arizona (November 2006) 54–67
6. Krogstie, J.: Integrating the understanding of quality in requirements specification and conceptual modeling. SIGSOFT Softw. Eng. Notes **23**(1) (1998) 86–91
7. Codd, E.F.: Normalized data structure: A brief tutorial. In: SIGFIDET Workshop, ACM (1971) 1–17
8. Codd, E.F.: Further normalization of the data base relational model. IBM Research Report, San Jose, California **RJ909** (1971)
9. Kent, W.: A simple guide to five normal forms in relational database theory. Commun. ACM **26**(2) (1983) 120–125
10. Moody, D.: Dealing with "map shock": A systematic approach for managing complexity in requirements modelling. REFSQ (2006)
11. Moody, D.: What makes a good diagram? improving the cognitive effectiveness of diagrams in is development. In Knapp, Magyar, eds.: Intl Conf on Information Systems Development, Budapest, Hungary, Springer (August 31-2 2006)
12. Dullea, J., Song, I.Y., Lamprou, I.: An analysis of structural validity in entity-relationship modeling. Data Knowl. Eng. **47**(2) (2003) 167–205
13. Boufares, F., Bennaceur, H.: Consistency problems in er-schemas for database systems. Inf. Sci. **163**(4) (2004) 263–274
14. Damm, F.M., Hansen, B.S., Bruun, H.: On type checking in vdm and related consistency issues. In: VDM '91: Proceedings of the 4th International Symposium of VDM Europe on Formal Software Development-Volume I, London, UK, Springer-Verlag (1991) 45–62
15. Chen, P.P.S.: The entity-relationship model—toward a unified view of data. ACM Trans. Database Syst. **1**(1) (1976) 9–36
16. Hoxmeier, J.A.: Typology of database quality factors. Software Quality Journal **7**(3/4) (1998) 179–193
17. Kesh, S.: Evaluating the quality of entity relationship models. Information and Software Technology **37**(12) (dec 1995) 681–689
18. Briand, L.C., Morasca, S., Basili, V.R.: Property-based software engineering measurement. IEEE Trans. Softw. Eng. **22**(1) (1996) 68–86
19. Habra, N., Abran, A., Lopez Martin, M.A., Sellami, A.: A framework for the design and verification of software measurement methods. Journal of Systems and Software **81**(5) (2008) 633–648
20. Nelson, R.R., Todd, P.A., Wixom, B.H.: Antecedents of information and system quality: An empirical examination within the context of data warehousing. Journal of Management Information Systems **21**(4) (2005) 199–235
21. Lindland, O.I., Sindre, G., Solvberg, A.: Understanding quality in conceptual modeling. IEEE Softw. **11**(2) (1994) 42–49
22. Moody, D.L., Sindre, G., Brasethvik, T., Sølvberg, A.: Evaluating the quality of information models: empirical testing of a conceptual model quality framework. In: ICSE '03: Proceedings of the 25th International Conference on Software Engineering, Washington, DC, USA, IEEE Computer Society (2003) 295–305
23. Serrano, M., Trujillo, J., Calero, C., Piattini, M.: Metrics for data warehouse conceptual models understandability. Inf. Softw. Technol. **49**(8) (2007) 851–870

24. Manso, M.E., Genero, M., Piattini, M.: No-redundant metrics for uml class diagram structural complexity. In Eder, J., Missikoff, M., eds.: CAiSE. Volume 2681 of Lecture Notes in Computer Science., Springer (2003) 127–142
25. Genero, M., Piattini, M., Manso, M.E.: Finding "early" indicators of uml class diagrams understandability and modifiability. In: ISESE, IEEE Computer Society (2004) 207–216
26. Bansiya, J., Davis, C.G.: A hierarchical model for object-oriented design quality assessment. IEEE Trans. Software Eng. **28**(1) (2002) 4–17
27. Genero, M., Piattini, M., Manso, M.E., Garcia, F.: Early metrics for object oriented information systems. In: OOIS '00: Proceedings of the 6th International Conference on Object Oriented Information Systems, London, UK, Springer-Verlag (2000) 414–425
28. Cherfi, S.S.S., Akoka, J., Comyn-Wattiau, I.: Perceived vs. measured quality of conceptual schemas: An experimental comparison. In: ER (Tutorials, Posters, Panels & Industrial Contributions), Australian Computer Society (2007) 185–190
29. Piattini, M., Calero, C., Sahraoui, H., Lounis, H.: Object-relational database metrics (2003)
30. Mens, T., Demeyer, S., eds.: Software Evolution. Springer (2008)
31. Wahler, M.: Using Patterns to Develop Consistent Design Constraints. PhD thesis, ETH Zurich, Switzerland (2008)
32. Eick, C.F.: A methodology for the design and transformation of conceptual schemas. In: VLDB '91: Proceedings of the 17th International Conference on Very Large Data Bases, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (1991) 25–34
33. Rauh, O., Stickel, E.: Standard transformations for the normalization of er schemata. In: CAiSE, Springer (1995) 313–326
34. Assenova, P., Johannesson, P.: Improving quality in conceptual modelling by the use of schema transformations. In: ER '96: Proceedings of the 15th International Conference on Conceptual Modeling, London, UK, Springer-Verlag (1996) 277–291
35. Hainaut, J.L.: The transformational approach to database engineering. In Lämmel, R., Saraiva, J., Visser, J., eds.: Generative and Transformational Techniques in Software Engineering. Volume 4143 of LNCS., Springer (2006) 95–143
36. OMG: UML 2.0 OCL Final Adopted Specification. OMG (2003)